# DANTE – The combination between an Ant Colony Optimization algorithm and a Depth Search method

Pedro Cardoso – Mário Jesus
EST – Universidade do Algarve
8005-139 Faro, Portugal
pcardoso-mjesus@ualg.pt

Alberto Marquez
Dept. Matemática Aplicada I
ETSII – Universidad de Sevilla
Avda. Reina Mercedes
s/n 41012 Sevilla, Spain
almar@us.es

## Abstract

*The $\epsilon$-DANTE method is an hybrid meta-heuristic. In combines the evolutionary Ant Colony Optimization (ACO) algorithms with a limited Depth Search. This Depth Search is based in the pheromone trails used by the ACO, which allows it to be oriented to the more promising areas of the search space. Some results are presented for the multiple objective $k$-Degree Spanning Trees problem, proving the effectiveness of the method when compared with other already tested evolutionary methods.*

## 1. Introduction

It is well known that, very often, the use of pure Evolutionary Algorithms has a lack of performance, namely in the optimization of large combinatorial problems instances. Moreover, the majority of the evolutionary meta-heuristics have some common characteristics. For example, they usually reach good approximations to the optimum solutions (despite the difficulties to refine those approximations, i.e., getting to the real optimum). Often only the best solution(s) are kept. The other solutions are discarded without further exploration, which forgets the expensive computational effort necessary to build them. Examples of top performing meta-heuristics that employ this methodology are the Simulated Annealing, the Tabu Search, or the Ant Colony Optimization algorithms [11]. Therefore, this strategy does not allow a proper local exploration of the eventually more promising regions of the search space. In other words, the neighborhoods of the obtained solutions do not go through an exhaustive exploration.

Algorithmic hybridizations appeared as an effort to solve some of the above mentioned problems. The improvement of the methods by those combinations can give us the best of several algorithmic strategies. In the implemented cases, different possible hybridizations can be thought, like the use of initial solutions via another method than the main one (used many times for instance in the Genetic Algorithms to obtain the initial population[9, 11]) or a local improvement of the obtained solutions. Usually, hybrid methods are considered as the ones that use two or more methods in the following sense: the primary methods are used to generate more or less rough approximations to the problems solutions, followed by other(s) method(s) that refine the earlier solutions. Some examples of the application of the second phase to the solution obtained by some meta-heuristic, are the use of the 2-OPT or 3-OPT for the Travelling Salesman Person [5], the SOP-3-exchange for the Sequential Ordering Problem [6], or the Iterated Local Search for the Bin Packing Problem [10]. In [1] is presented the Beam–ACO, which is a combination of a Beam Search heuristic with an ACO algorithm. In this case, the solution construction mechanism of standard ACO algorithms is replaced by the solution construction mechanism in which each artificial ant performs a probabilistic Beam Search. This is done by replacing the deterministic choice of a solution component at each construction step by a probabilistic choice based on transition probabilities.

In this paper, we propose a hybrid metaheuristic called $\epsilon$–Depth ANT Explorer ($\epsilon$-DANTE), that uses an efficient local search (adapted to a pheromone-oriented procedure). More precisely, in each one of the $\epsilon$-DANTE cycles, sets of solutions are computed using a mandatory first phase that, in some cases, is followed by an elective second phase that depends on the quality of the former solution. More specifically,

- In the first phase, similar to most ACO algorithm, a solution is generated based in a constructive procedure that successively adds selected components according to a pseudo-random formula.

978-0-7695-3326-1/08 $25.00 © 2008 IEEE
DOI 10.1109/HIS.2008.115

- In the second phase, a limited depth search procedure based on the best fitting solutions is made. In other words, if the first phase outcome is within $\epsilon$ range to the approximation set (the set of the best known performing solutions) or improves this set, then a limited depth search procedure is applied. The same pheromone trails, that were used in the first phase, are also used here. This enhances the limitations of the depth search methods by leading the procedure to include the more promising components in the constructed solutions.

Therefore, this paper is structured as follows. The $\epsilon$-DANTE is described in the next section. In the third section are presented some results and, in the last one, are drawn some conclusions.

## 2. $\epsilon$–DANTE – Depth ANT Explorer

The second phase local searches mentioned above (e.g., 2-OPT, 3-OPT, ...) are used to refine the solutions, but do not use the information acquired by the evolutionary metaheuristic, except for the base solution itself. Moreover, most of the times the local search operator are applied to all solutions even in the cases where they are not much promising.

To try to explore some of these weaknesses we developed the $\epsilon$–Depth ANT Explorer ($\epsilon$–DANTE) which is a hybrid method. In fact, $\epsilon$–DANTE is little more since, it is a fusion between an Ant Colony Optimization method [4] and an Depth Search method [8]. From this combination results more than a meta-heuristic followed by the local search, as we will see next.

The kernel of the method, similarly to the Ant Colony Optimization method, has a set of cycles. In each cycle a set of solutions is built using the pheromone trails. Those solutions are themselves used to update the pheromone trails and to update the set of (best) approximations to the problem solution: a single set for the single objective problems or the Pareto set for the multiple objective case [3] (see Section 3). Alternatively, the updating of the pheromone trails can be done in a more greedy method using only the best solutions that where obtained during the entire process [2].

One of the main ideas of the $\epsilon$-DANTE, is to make a local exploration of the best solutions. In other words, a solution is attractive whenever certain quality pattern is achieved. This can be quantified has the solution being within an $\epsilon$ range from the best known solutions. In this case, the Depth Search based on the earlier built solution is applied. This strategy tries to avoid the, probably useless, computational effort associated with the exploration of neighborhoods of the worst solutions. The Depth Search is limited both in the depth itself and in the number of possible branches of the search tree. Furthermore, the depth search is oriented by the pheromone trails and the local heuristics used in the Ant Colony method (Section 2.1 describes in more detail the Depth Search phase).

1. Initialize the pheromone trail.
2. **While** stopping criterion is not met do
   (a) **For all** ants **do**
   - Construct a new solution, S, using the current pheromone trail.
   - **If** the distance of S to the approximation set is inferior to $\epsilon$ or S improves the approximation set **then** apply a limited depth search procedure, based on the pheromone trails, from that solution.

   (b) Update the pheromone trail

**Figure 1. $\epsilon$-DANTE Algorithm**

The main $\epsilon$-DANTE algorithm is sketched in Figure 1.

### 2.1 Depth Search phase

As referred, the fitness of each generated solution, $S$, is compared with the fitness of the elements in the approximation set, $\mathcal{P}$ ($\mathcal{P}$ is the approximation set: in the single objective case it will be a singular set and in the multiple objective case it will have the non-dominated solutions [3] – see section 3.1). Then the limited depth search goes to

**Level $D$ –** If $S$ improves $\mathcal{P}$, that is, $S$ is not worst than any element of $\mathcal{P}$; or

**Level $d$ –** If $S$ is worst than some element of $\mathcal{P}$, but its relative distance to the elements of the approximation set is smaller than $\epsilon$.

Here $d$ and $D$ are algorithm parameters and should verify $d > D$. Furthermore, the number of branches used in each level of the Depth Search, $M$, is also an algorithm parameter.

Figure 2 presents a high level description of the process where `Function` $\epsilon$-DANTE_`Solution` requires some parameters besides the solution in construction $T$, namely

- $\epsilon$ – The $\epsilon$ parameter will influence the number of times that the process enters in the Depth Search mode. Small values will guarantee that only the solutions with objective value near the best known solutions will go into the oriented Depth Search. Larger values will do the opposite.
- The algorithm contains a tabu list for each level, $tabu_l$, that is initialized as empty set. This tabu list avoids

**Function** $\epsilon$-DANTE_Solution($T$)
$l \leftarrow |T|$ */*defines the level by the number of added components*/*
**if** $T$ is a solution **then**
Set $\Delta$ as the relative distance from $T$ to $\mathcal{P}$
**if** $\Delta < 0$ **then** */*$T$ improves $\mathcal{P}$*/*
Update the approximation set with $T$
Update $\Delta_k$ $(k = 1, 2, \ldots, m)$ with $T$ †
**return** $D$
**else**
**if** $\Delta < \epsilon$ **then**
Update $\Delta_k$ $(k = 1, 2, \ldots, m)$ with $T$ †
**return** $d$
**else return** 0
**else**
$slevel \leftarrow 0$
$NBranches \leftarrow M$
**for** $k = 1$ **to** $NBranches$
**if** $\exists_{e \in \mathcal{E} - tabu_l}$ : $T \cup \{e\}$ is admissible **then**
Choose an edge, $e$, from $\mathcal{E} - tabu_l$ ‡
$Tabu_l \leftarrow Tabu_l \cup \{e\}$ †
$T \leftarrow T \cup \{e\}$
*/*Recursive call*/*
$L \leftarrow \epsilon$-DANTE_SOLUTIONS($T$)
**if** $L > 0$ **then**
$NBranches \leftarrow NBranches + M$
$slevel \leftarrow \max\{slevel, L\}$
$T \leftarrow T - \{e\}$
**else**
**break** */*for*/*
$Tabu_l \leftarrow \emptyset$ † */*clean the tabu list*/*
**return** $\max\{slevel - 1, 0\}$

**Figure 2. $\epsilon$-DANTE's solutions exploration algorithm (where the steps marked with † are optional and the ones with ‡ are problem specific).**

that the same solutions are rebuilt in the Depth Search, by restricting the addition of the same componets in the same level. The use of the tabu list is implementation dependent and should be thought according to the cleaning $tabu_l$ step, since it also restricts the construction of new solutions containing those components.

- The computation of the distance from $T$ to the approximation set $\mathcal{P}$ returns a non-positive value if $T$ improves $\mathcal{P}$;

- The update procedure consists in inserting $T$ in $\mathcal{P}$ and removing the elements of $\mathcal{P}$ that are worst than $T$.

- The update of $\Delta_k$ is optional since, if all solutions contribute to the variation of the pheromones, the result is a noisy trail with consequent lost of performance. Alternative strategies include a greedy update where only the best solutions contribute to the variation of the pheromone trails, or even a more restricted selection over the approximation set (see section 3.4).

A more detailed description of the method can be found in [2].

## 3 Some computational tests

### 3.1 Test problems - Multiple Objective $k$-Degree Minimum Spanning problem

To test the $\epsilon$-DANTE method it was considered the Multiple Objective $k$-Degree Minimum Spanning problem. A $k$-Degree Minimum Spanning Tree is a minimal weight spanning tree such that the maximum degree of any node is $k$. The tree is built over a network $(\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V}$ is the nodes set, $\mathcal{E}$ is the edges set, $\mathcal{W} : \mathcal{E} \rightarrow \mathbb{R}^m$ is the weight vector-function defined as $\mathcal{W}(e) = (w_1(e), w_2(e), \ldots, w_m(e))$, and $m$ is the number of objectives.

Here, since we have a multiple objective problem, minimal weight is considered in the Pareto optimality context. That is, given two solutions $X$ and $Y$ of the feasible set $\mathcal{S}$, it is said that $X$ dominates $Y$, $X \prec Y$, if for all $i \in \{1, 2, \ldots, m\}$ we have $w_i(X) \leq w_i(Y)$, and exists $j \in \{1, 2, \ldots, m\}$ such that $w_j(X) < w_j(Y)$. A solution $X$ is Pareto (optimal) if it is not dominated by any other solutions of $\mathcal{S}$, that is, for all $Y \in \mathcal{S} - \{X\}$ it is verified that $Y \not\prec X$. The set of all Pareto solutions is called Pareto set or efficient set.

### 3.2 Metrics

To verify the performance our method we used two metrics: $R1$ and $R3$ [7]. The $R1$ metric measures the probability that an approximation set $\mathcal{P}_1$ is better than another $\mathcal{P}_2$ over a family of utility functions $U$. If $R1(\mathcal{P}_1, \mathcal{P}_2, U) > \frac{1}{2}$ then, according to this measure, $\mathcal{P}_1$ is better than $\mathcal{P}_2$ and it will be not worse if $R1(\mathcal{P}_1, \mathcal{P}_2, U, p) \geq \frac{1}{2}$. The $R3$ metric measures the expected proportion of superiority of one set over another. The larger $R3(\mathcal{P}_1, \mathcal{P}_2, U)$ is, the worst is $\mathcal{P}_1$ when compared with $\mathcal{P}_2$. The opposite is valid if $R3$ is negative. If $R3(\mathcal{P}_1, \mathcal{P}_2, U) \approx 0$ then $\mathcal{P}_1$ and $\mathcal{P}_2$ have similar quality over $U$.

### 3.3 $k$-Degree Spanning Trees problem construction

In the proposed process, to build the $k$-degree spanning tree, feasible edges are successively added until a solution is complete. Before the addition of any edge, it must be verified that such addition will not form a cycle neither the nodes maximum degree condition is violated.

More precisely, the process starts by randomly selecting a node from $\mathcal{V}$, $s$, and setting $T_N = \{s\}$ ($T_N$ is the set of the nodes already included it the tree). Then $n-1$ admissible edges from $\mathcal{A} = \{e_{uv} \in \mathcal{E} : u \in T_N \wedge \delta(u) < k \wedge v \in \mathcal{V} - T_N\}$, are sequentially added, where $\delta$ is the degree of node $u$ in the subtree that is being constructed, and $k$ the maximum degree allowed. The selection of the edges is pseudo-randomly made using formula

$$e_{st} = \begin{cases} \arg\max_{f \in \mathcal{A}} \prod_{j=1}^{m} \tau_j(f)^{\alpha_j} w_j(f)^{\beta_j} & \text{if } q \leq q_0 \\ e & \text{if } q > q_0, \end{cases} \quad (1)$$

where

- $\tau_j(e)$ is the pheromone value associated to the $j$ weight in edge $e$;
- $w_j(e)$ is the $j$-weight of edge $e$;
- $\alpha_j$ is an algorithm parameter associated to the relevance of weight $j$;
- $\beta_j$ is an algorithm parameter associated to the local heuristic that favours edges with lower $j$-weight;
- $e \in \mathcal{A}$ is an edge pseudo-randomly chosen with probability

$$p(e) = \frac{\prod_{j=1}^{m} \tau_j(e)^{\alpha_j} w_j(e)^{-\beta_j}}{\sum_{f \in \mathcal{A}} \prod_{j=1}^{m} \tau_j(f)^{\alpha_j} w_j(f)^{-\beta_j}}. \quad (2)$$

- $q$ is a uniform random value in $[0, 1]$; and
- $q_0 \in [0, 1]$ is a parameter that influences which branch of (1) is used more often: a smaller value of $q_0$ produces a more exploratory search, since it implies the use of the pseudo-random formula (2) with higher probability. When $q_0$ is near 1, the feasible edge with larger probability of entering the tree is used with greater frequency, which suggest an exploiting search.

If the solution satisfies the $\epsilon$ distance constraint then the process enters Depth Search method described in Algorithm 2. The selection of the edges to enter the search tree in this phase also follows the above formulas, that is, (1) and (2).

### 3.4 Pheromone update

To update the pheromones matrices it was used the Angle-Pheromone Update strategy [2]. This strategy can be considered greedy in the sense that it only uses elements of the approximation set. The objective is to explore small regions of the search space by using, in the pheromone updating formula, only a subset of the solutions contained in that set. This idea is motivated by the fact that, in most of the cases, the number of elements in the approximation set becomes very large. Empirical tests proved that if all the solutions in the approximation set were used, the pheromone based selection becomes very noisy, which delays the convergence toward the Pareto set. For the single objective case, a set of the $k$ best solutions can be kept, using a subset of that set to do pheromones update.

Therefore, the pheromone vector update is made according to formula

$$\tau(e) = \rho\tau(e) + \Delta(e), e \in \mathcal{E},$$

where

- $\tau(e)$ is a pheromone vector associated to edge $e$;
- $\rho \in [0, 1]$ is called the persistence factor ($1 - \rho$ is the evaporation factor). The smaller the values of $\rho$ are, the smaller quantity of information, used in one cycle, is transmitted to following cycle;
- $\Delta(e) = (\Delta_1(e), \Delta_2(e), \ldots, \Delta_m(e))$ is the reinforcement pheromone vector associated to edge $e$ and is computed using the elements of the approximation set, $\mathcal{P}$, and formula

$$\Delta_k(e) = \sum_{T \in \mathcal{P}_e} \frac{Q}{w_k(T)},$$

where

  - $Q$ is a value with the same magnitude of the solutions. For example, if the weights are balanced it can be used the average of the minimum weights,

$$\frac{1}{m} \sum_{k=1}^{m} \min_{T \in \mathcal{P}} w_k(T);$$

  - $\mathcal{P}_e$ are the elements of the approximation set that contain edge $e$ and lie in a subangle of the angle defined by the origin and the extreme solutions (of weights $k$ and $k+1$), that is,

$$\mathcal{P}_e = \{T \in \mathcal{P} : e \in T \wedge \gamma_1 \leq \phi_k(T) \leq \gamma_2\}$$

where

$$\phi_k(T) = \arctan\left(\frac{w_{k+1}(T)}{w_k(T)}\right),$$

| Parameter | Values |
|---|---|
| $D$ | 1 |
| $d$ | $\lfloor \frac{3\|\mathcal{V}\|}{4} \rfloor$ |
| $M$ | 2 |
| $\epsilon$ | 0 |
| $\alpha_i$, $\beta_i$ | $\{0, 0.03, 0.06, \ldots, 3.0\}$ |
| $\rho$ | 0.1 |
| $q_0$ | $\{0.5, 0.6, 0.7, 0.8, 0.9\}$ |
| $k$ | 3 |
| Number of ants per cycle | $\|\mathcal{V}\|$ |
| Number of cycles | 2 |
| Maximum run time | $\min\{60\|\mathcal{V}\| \log \|\mathcal{V}\|, 36000\}$ Seconds |

**Table 1. Used parameters.**

$$\begin{cases} \gamma_1 &= \phi_k^{\min} + I_k \phi_k^h \\ \gamma_2 &= \phi_k^{\min} + (I_k + 1)\phi_k^h \end{cases},$$

$$\begin{cases} \phi_k^{\min} &= \min_{T \in \mathcal{P}} \phi_k(T) \\ \phi_k^{\max} &= \max_{T \in \mathcal{P}} \phi_k(T) \end{cases},$$

for $k = 1, 2, \ldots, m-1$, $\phi_k^h$ is the step interval, and $I_k$ is a value related to the region to be explored which is controlled by the main process.

### 3.5 Results

The algorithm $\epsilon$-DANTE was implemented in C++ and tests were run on a PC with a $3Ghz$ Intel Pentium IV$^{\text{TM}}$processor, $512Mb$ of RAM and Windows XP OS. For each problem the method was run 15 times with the parameters reported in Table 1.

The problems instances that were used to test the implementation were defined by [9] and the $\epsilon$-DANTE solutions are compared with the solutions obtained with the Genetic Algorithms presented by the same authors. More specifically, it was used as reference set the one composed by the non-dominated elements of the union of the 30 approximation sets presented in [9].

Table 2 presents a resume of the results for some of the tested instances over the 15 runs. This table contains information about the average values of the metrics $R1$ and $R3$, the average cardinality of the approximation set built with $\epsilon$-DANTE (in parentheses the cardinal of the reference set), and the average time of the last update of the approximation set. From the same table it is possible to observe that in most of the cases, $\epsilon$-DANTE improves the reference set since $R1 < 0.5$ and $R3 \approx 0$. The exceptions were the 10 nodes networks where in one case it has achieved the exact Pareto set (obtained with the Brute Force method) and in the other two cases it has obtained 128 and 183 of the 134 and 191 solutions, respectively.

| Problems | $\mu_{R1}$ | $\mu_{R3}$ | $\mu_{\lvert\mathcal{P}\rvert}$ $(\mu_{\lvert\mathcal{P}_{ref}\rvert})$ | $\mu_{Time}$ |
|---|---|---|---|---|
| 10vac | 0.57 | 0.00 | 183 (191$^{\dagger}$) | 7.6 |
| 10v-m-c | 0.5 | 0.00 | 129 (129$^{\dagger}$) | 628 |
| 10vconc | 0.54 | 0.00 | 128 (134$^{\dagger}$) | 8 |
| 25vac | 0.47 | 0.00 | 517 (439) | 3754 |
| 25vc | 0.39 | 0.00 | 2496.6 (1480) | 4351. |
| 25v-m-c | 0.13 | 0.01 | 2168.7 (820) | 4743 |
| 50vac | 0.40 | -0.05 | 6328.1 (1436) | 11960 |
| 50vconc1 | 0.48 | 0.00 | 1748.6 (894) | 11803 |

**Table 2. Resume of the results for the $k$-Degree Minimum Spanning Tree problem.**

The time evolution of the $R1$ and $R3$ metrics are depicted in Figures 3 and 4, respectively, for some of the tested instances (25vac, 25vc and 50vac). It is possible to observe that the convergence of the solutions toward the reference set is made quite fast since $R3$ quickly takes values near to zero. Nevertheless, the local refinement takes some extra time, as reflected in the last update time of the approximation set ($\mu_{Time}$), which as already referred is a characteristic common to most of the Ant Colony based algorithms.

## 4 Conclusions

This paper was devoted to the study of the $\epsilon$-DANTE method. This method is based in the Ant Colony Optimization paradigm and appears as an effort to provide a more effective way to further exploring the best fitness performing solutions. More precisely, whenever a solution is inserted into the approximation set, or satisfies an $\epsilon$ distance to the approximation set criterion, it is performed a limited Depth Search using the pheromone values to guide the search.

Results for a version of $\epsilon$-DANTE applied to the Multiple Objective $k$-Degree Minimum Spanning Trees are presented. It was verified that the method rapidly converges toward the fronts achieved by other authors and ends up by improving, in general, their results.

As in many Ant Colony algorithms, the $\epsilon$-DANTE uses a greedy strategy, where only the best performing solutions are used to update the pheromone value.

## References


[1] C. Blum. Beam–ACO – hybridizing ant colony optimization with beam search: an application to open shop scheduling. *Computers and Operations Research*, 32(6):1565–1591, 2005.

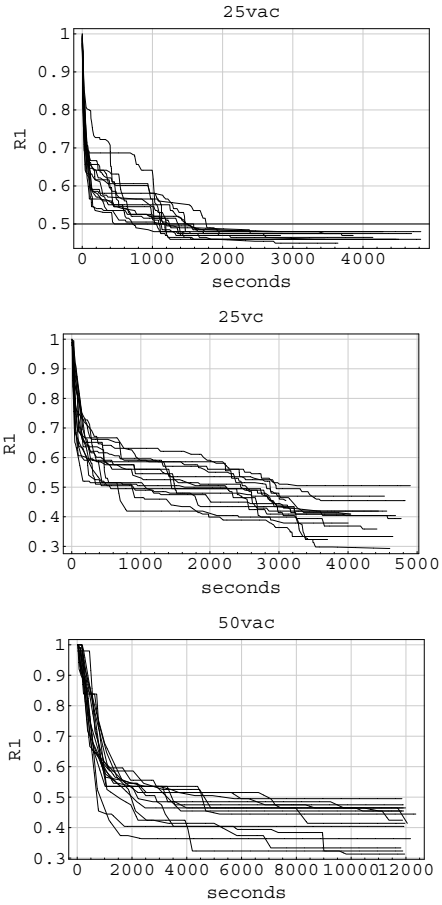


**Figure 3. Time evolution for $R1$ metric over the 15 runs.**

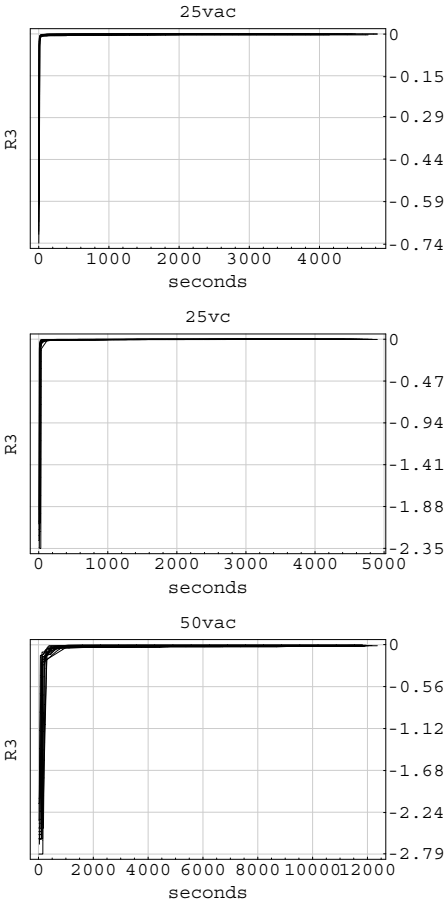


**Figure 4. Time evolution for $R3$ metric over the 15 runs.**


[2] P. Cardoso. *Ant Colony Algorithms for Multiple Objective Combinatorial Optimization: Applications to the Minimum Spanning Trees Problems*. PhD thesis, University of Seville, Spain, Mar 2007.

[3] K. Deb. *Multi-objective Optimization using Evolutionary Algorithms*. John Wiley & sons, 2001.

[4] M. Dorigo, E. Bonabeau, and G. Theraulaz. *Swarm Intelligence: From Natural to artificial Systems*. Oxford University Press, 1999.

[5] M. Dorigo and T. Stutzle. *Ant Colony Optimization*. MIT Press, 2004.

[6] L. Gambardella and M. Dorigo. An ant colony system hybridized with a new local search for the sequential ordering problem. *INFORMS, Journal on Computing*, 12(3), 2000.

[7] A. Jaszkiewicz. *Multiple Objective Metaheuristic Algorithms for Combinatorial Optimization*. PhD thesis, Poznan University of Technology, 2001.

[8] D. Jungnickel. *Graphs, Networks and Algorithms*, volume 5 of *Algorithms and Computation in Mathematics*. Springer-Verlag, Berlin, 1999.

[9] J. Knowles and D. Corne. Benchmark problem generators and results for the multiobjective degree-constrained minimum spanning tree problem. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 424–431. Morgan Kaufmann Publishers, San Francisco, California, 2001.

[10] J. Levine and F. Ducatelle. Ant colony optimisation and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society, Special Issue on Local Search*, 55(7), 2004.

[11] I. Parmee. *Evolutionary and Adaptive Computing in Engineering Design*. Springer-Verlag, London, 2001. ISNB:1-85233-029-5.

[12] J. Puchinger and G. R. Raidl. Combining metaheuristics and exact algorithms in combinatorial optimization: A survey and classification. In *Proceedings of the First International Work-Conference on the Interplay Between Natural and Artificial Computation*, volume 3562 of *LNCS*. Springer, 2005.

[13] G. R. Raidl. A unified view on hybrid metaheuristics. In *Proceedings of the Hybrid Metaheuristics Workshop*, number 4030 in LNCS. Springer, 2006.